#### \*\*income is colored in green; expenses colored in red

*\*\* Pam added notes at bottom with gathered hypotheses and ideas, 4/11/20, 10:30am*

*\*\*Nicole added notes, 4/12/20 8 PM*

*\*\*Cat’s rating analysis in orange*

#### **2. Assumptions**

Based on research completed prior to launching App Trader as a company, you can assume the following:

\*\*Pam - checking out averages, mean, medians/exploratory data analysis

* *Summarize price info by genre for app\_store\_apps*
* SELECT primary\_genre,
* MIN(price),
* ROUND(AVG(price),2) AS avg\_price,
* ROUND(PERCENTILE\_CONT(0.50) WITHIN GROUP
* (ORDER BY price)::numeric, 2) AS median\_price,
* MAX(price)
* FROM app\_store\_apps
* GROUP BY primary\_genre;
* *Summarize price info by genre for play\_store\_apps*
* SELECT DISTINCT genres,
* MIN(to\_number(price, 'G999D99')),
* ROUND(AVG(to\_number(price, 'G999D99')),2) AS avg\_price,
* PERCENTILE\_CONT(0.50) WITHIN GROUP
* (ORDER BY to\_number(price, 'G999D99')) AS median\_price,
* MAX(to\_number(price, 'G999D99'))
* FROM play\_store\_apps
* GROUP BY genres;

a. App Trader will purchase apps for 10,000 times the price of the app. For apps that are priced from free up to $1.00, the purchase price is $10,000.

b. Apps earn $5000 per month on average from in-app advertising and in-app purchases *regardless* of the price of the app.

c. App Trader will spend an average of $1000 per month to market an app *regardless* of the price of the app. If App Trader owns rights to the app in both stores, it can market the app for both stores for $1000 per month. - Nicole - this is similar to e), find lists that should be excluded? Or ideas to why an app should be included if it’s only on one store

*\*\*If an app is on both stores, App Trader can get double the marketing power on both platforms for the same $1000/month*

d. For every half point that an app gains in rating, its projected lifespan increases by one year, in other words, an app with a rating of 0 can be expected to be in use for 1 year, an app with a rating of 1.0 can be expected to last 3 years, and an app with a rating of 4.0 can be expected to last 9 years. Ratings should be rounded to the nearest 0.5 to evaluate its likely longevity. - Cat - generate a list of apps for each store and their calculated life span

e. App Trader would prefer to work with apps that are available in both the App Store and the Play Store since they can market both for the same $1000 per month. jacob - find which apps are on both stores, and list of apps that are on one and not the other, vice versa so we know which ones to exclude

#### **3. Deliverables**

a. Develop some general recommendations as to price range, genre, content rating, or anything else for apps that the company should target.

b. Develop a Top 10 List of the apps that App Trader should buy next week for its Memorial Day debut.

c. Prepare a 5-10 minute presentation for the leadership team of App Trader to inform them of your recommendations.

### **All analysis work must be done in PostgreSQL, however you may export query results if you want to create charts in Excel for your presentations**

NOTES FROM PAM:

I gathered info from my family about what they look for in apps, and here are some other hypotheses and ideas to look at:

* Price: definitely < $10.00, but especially <=$3.99
* If install\_count range is “good” (whatever we determine that is) in the play\_store\_apps and the app is in both stores, it’s likely to be in the appropriate range for both stores. Look for AVG(install\_count), use middle to bottom numbers, don’t want apps that have already been installed millions of times (unless maybe high ratings with long expected life -- see assumptions). Low install\_count with high ratings might indicate newer released apps.
* Check Minecraft, Google Photos, Chrome for ideas about sustainability (high install\_count but long life)
* Categories to look at: dating, games, productivity, maps, news/weather, browsers, cooking/recipes, streaming/movies/netflix/hulu *Agree! I am working on some queries to compare genres and proportions with review\_count/install\_count*
* *From Pam 4/13, 7:30pm*
* *--Explore app genres and categories from each db.*
* *-- 23 rows in Apple store, 119 in Android store*
* *-- (NOTE: Android genres offer subcategories using genres,*
* *-- wider genre is covered by Category field info, 33 categories available)*
* *SELECT DISTINCT primary\_genre*
* *FROM app\_store\_apps;*
* *SELECT DISTINCT genres*
* *FROM play\_store\_apps;*
* *SELECT DISTINCT category*
* *FROM play\_store\_apps;*
* Kids are doing the majority of downloading apps, so keep content\_rating mostly PG
* Check out app on app\_store\_apps that allows access to iphone apps for non-iphone devices. Android devices/tablets also have app that acts like PC emulator.
* iphone/apple devices more popular (according to my daughter), so those apps more popular?
* Iphones cost more, so users with those devices have more $ to spend on apps?
* Get count of ratings:

(only an idea, doesn’t work yet)

SELECT content\_rating, count(\*)

FROM app\_store\_apps

LEFT JOIN

SELECT content\_rating, count(\*)

FROM play\_store\_apps

GROUP BY name

HAVING count(\*) > 100 (or whatever # we use here);

(NOTE - need a join here)

MINOR thoughts

* Hubby said to look for apps with regular support updates and recent release\_dates, but we don’t have that info here. Just FYI, might be useful in presentation.

**Trying to classify apps by rating and “Lifespan” as a place to start making some determinations. This isn't perfect yet!**

RUN THIS WHEN YOU CAN AND SEE WHAT YOU THINK!

SELECT genres, name, rating, install\_count,

CASE WHEN rating > 3.0 then 'high rating'

WHEN rating < .5 then 'low rating'

ELSE 'medium rating'

END AS rating\_category

FROM play\_store\_apps

WHERE rating is NOT NULL

GROUP BY genres, name, rating, install\_count

ORDER BY install\_count;

UPDATED BUT WITHOUT CAST for install\_count --Help please?

Got it!! Sent to you in slack! -- pam 9:20pm 4/13

SELECT genres, name, rating, price, TRIM(install\_count, '+,') AS clean\_count,

CASE WHEN rating > 3.0 then 'high rating'

WHEN rating < .5 then 'low rating'

ELSE 'medium rating'

END AS rating\_category

FROM play\_store\_apps

WHERE rating is NOT NULL

GROUP BY genres, name, rating, price, clean\_count

ORDER BY rating DESC, clean\_count DESC;

--and for APP STORE \*\* I added price to this table\*\*

SELECT primary\_genre, name, rating, review\_count::numeric, price

CASE WHEN rating > 3.0 then 'high rating'

WHEN rating < .6 then 'low rating'

ELSE 'medium rating'

END AS rating\_category

FROM app\_store\_apps

WHERE rating is NOT NULL

GROUP BY primary\_genre, name, rating, review\_count, price

ORDER BY review\_count DESC;

Jacob

SELECT a.name, cast(a.review\_count as integer) + p.reviews as combined\_reviews,

round(a.rating, 1) as app\_rating, round(p.rating, 1) as play\_rating, a.price as app\_price,

p.price as play\_price

FROM app\_store\_apps as a

JOIN (SELECT name, max(review\_count) as reviews, rating, price

FROM play\_store\_apps

group by price, rating, name) as p

ON a.name = p.name

ORDER BY combined\_reviews desc;

Anyone want to try Purchase\_Price? I am close but can't make it run!

**Got it! Yay!**

SELECT name, price, (to\_number(price, 'G999D999') \* 10000) AS purchase\_price

FROM play\_store\_apps;

Nicole note: Here is purchase\_price for both stores in one table, but I need to figure out how to calculate the $0-0.99 ones, all of the others calculate! Working on this (4/13 8:30 PM)

**\*\*\*\* see next page, I fixed this and added some more! I think we have a list!!!**

SELECT app\_store\_apps.name AS app\_name, play\_store\_apps.name AS play\_name,

(CAST(app\_store\_apps.price as decimal) \*10000) AS app\_purchase\_price, --this does not calculate the $0-0.99 apps purchase price\*\*

(CAST(REPLACE(TRIM(play\_store\_apps.price), '$', '') AS decimal) \*10000) AS play\_purchase\_price

FROM app\_store\_apps LEFT JOIN play\_store\_apps

USING(name)

WHERE app\_store\_apps.name is not null

AND play\_store\_apps.name is not null

ORDER by play\_purchase\_price DESC

***\*\*\*\*\*\*\*TRY THIS ONE!!!! IF WE CAN DECIDE HOW TO ORDER IT (I HAVE RATING >4.5), WE MIGHT HAVE A LIST!!***

**(NOTE: I will say the apps at the top aren’t really what I expected, so we might need to look at this and see. It might be because of ratings instead of review\_count?):**

SELECT app\_store\_apps.name AS app\_name,

app\_store\_apps.primary\_genre,

play\_store\_apps.name AS play\_name,

play\_store\_apps.genres,

play\_store\_apps.rating AS ps\_apps\_rating,

app\_store\_apps.rating AS app\_apps\_rating,

CASE WHEN CAST(app\_store\_apps.price as money) <='0.99' THEN '$10,000'

WHEN CAST(app\_store\_apps.price as money) >'0.99'

THEN CAST(app\_store\_apps.price as money)\*10000

END AS app\_purch\_price,

CASE WHEN CAST(REPLACE(TRIM(play\_store\_apps.price), '$', '') AS money) <='0.99'

THEN '$10,000'

WHEN CAST(REPLACE(TRIM(play\_store\_apps.price), '$', '') AS money) >'0.99'

THEN CAST(play\_store\_apps.price as money)\*10000

END AS play\_purch\_price

FROM app\_store\_apps LEFT JOIN play\_store\_apps

USING(name)

WHERE app\_store\_apps.name is not null

AND play\_store\_apps.name is not null

AND play\_store\_apps.rating > 4.5

AND app\_store\_apps.rating > 4.0

ORDER by ps\_apps\_rating DESC, app\_store\_apps.rating DESC

LIMIT 25;

**Here’s the version using review\_count and it looks more like expected, but we need to weed out the duplicates using Jacob’s joined table, and find a way to add the install\_count cases from Cat’s work above, which I haven’t done yet:**

SELECT app\_store\_apps.name AS app\_name,

app\_store\_apps.primary\_genre,

play\_store\_apps.name AS play\_name,

play\_store\_apps.genres,

play\_store\_apps.review\_count AS ps\_apps\_review,

app\_store\_apps.review\_count AS app\_apps\_review,

play\_store\_apps.rating AS ps\_apps\_rating,

CASE WHEN CAST(app\_store\_apps.price as money) <='0.99' THEN '$10,000'

WHEN CAST(app\_store\_apps.price as money) >'0.99'

THEN CAST(app\_store\_apps.price as money)\*10000

END AS app\_purch\_price,

CASE WHEN CAST(REPLACE(TRIM(play\_store\_apps.price), '$', '') AS money) <='0.99'

THEN '$10,000'

WHEN CAST(REPLACE(TRIM(play\_store\_apps.price), '$', '') AS money) >'0.99'

THEN CAST(play\_store\_apps.price as money)\*10000

END AS play\_purch\_price

FROM app\_store\_apps LEFT JOIN play\_store\_apps

USING(name)

WHERE app\_store\_apps.name is not null

AND play\_store\_apps.name is not null

AND play\_store\_apps.rating > 4.5

ORDER by ps\_apps\_review DESC, ps\_apps\_rating DESC

LIMIT 25;

Show genres

***I think we just need to figure out how to show profit for our top 10 (might have to move to Excel if we can’t figure the math here).***

***PROFIT == (earnings/month \* projected lifespan of app) less (advertising/month \* projected lifespan of app) less purchase price***

***^^ maybe use AVG(earnings less advertising)??***

***So, ($5000/month\*??months) - ($1000/month\*??months) - purchprice == PROFIT***

***To figure projected lifespan of app, we use rating point system in assumptions.***

***\*\*make sure deliverables are: data-driven & follow readme***